

# Computação em Nuvem

GRADUAÇÃO EM REDES DE COMPUTADORES E  
ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

**Prof. Guto Muniz**

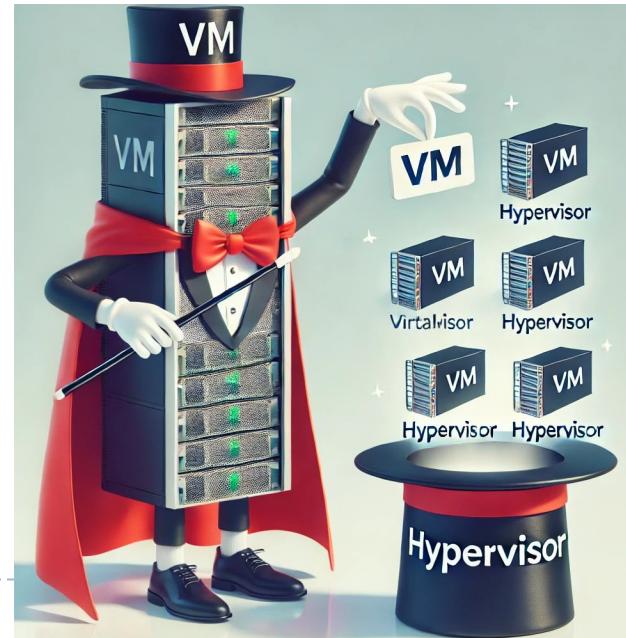
# Introdução

- Máquinas Virtuais (VMs)
- Docker
- Computação em Nuvem
- **Amazon Web Services (AWS)**



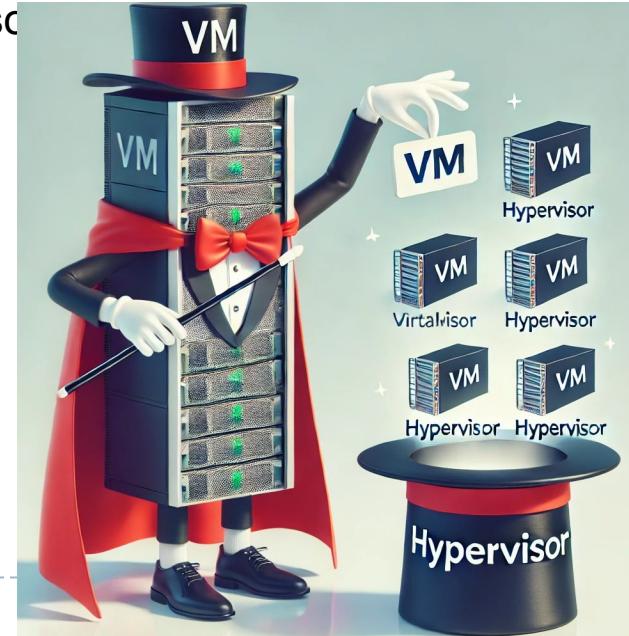
# Máquinas Virtuais (VMs)

- Virtualizar significa emular ou simular algo, é a capacidade de criar algo que não existe fisicamente.
- O que são Máquinas Virtuais?
  - Uma Máquina Virtual é um software que emula um computador físico, permitindo que múltiplos sistemas operacionais sejam executados simultaneamente em uma única máquina física.
  - VMs compartilham os recursos físicos do host, como CPU, memória e armazenamento, mas funcionam como sistemas independentes, com seus próprios sistemas operacionais e aplicativos.



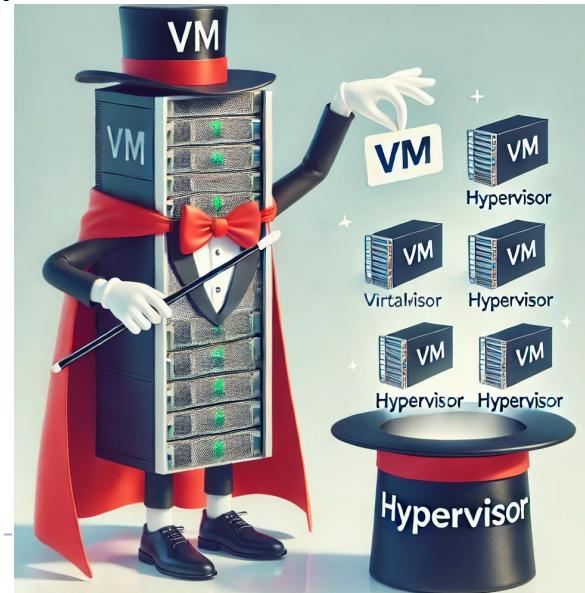
# Máquinas Virtuais (VMs)

- Tipos de VMs:
- **Tipo 1 (Bare Metal):**
  - Executado diretamente no hardware físico.
  - Oferece maior desempenho e controle sobre os recursos.
  - Exemplo: **VMware ESXi, Microsoft Hyper-V.**
- **Tipo 2 (Hosted):**
  - Executado sobre um sistema operacional já existente.
  - Mais fácil de configurar e usar, mas com desempenho potencialmente inferior ao Tipo 1.
  - Exemplo: **Oracle VirtualBox, VMware Workstation.**



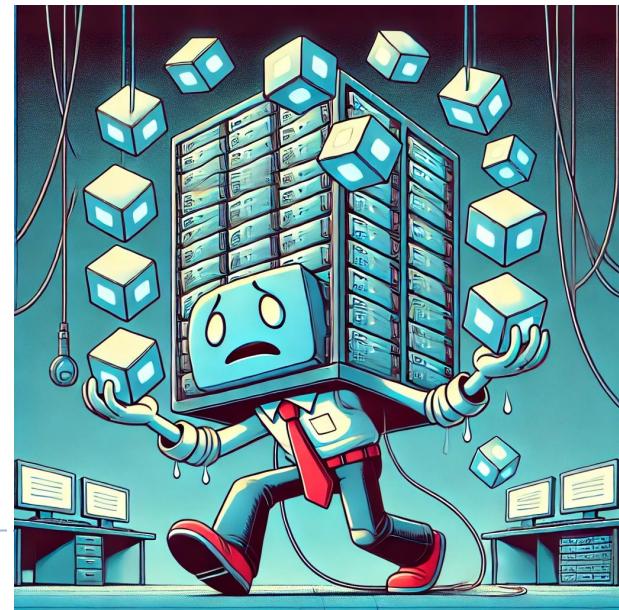
# Máquinas Virtuais (VMs)

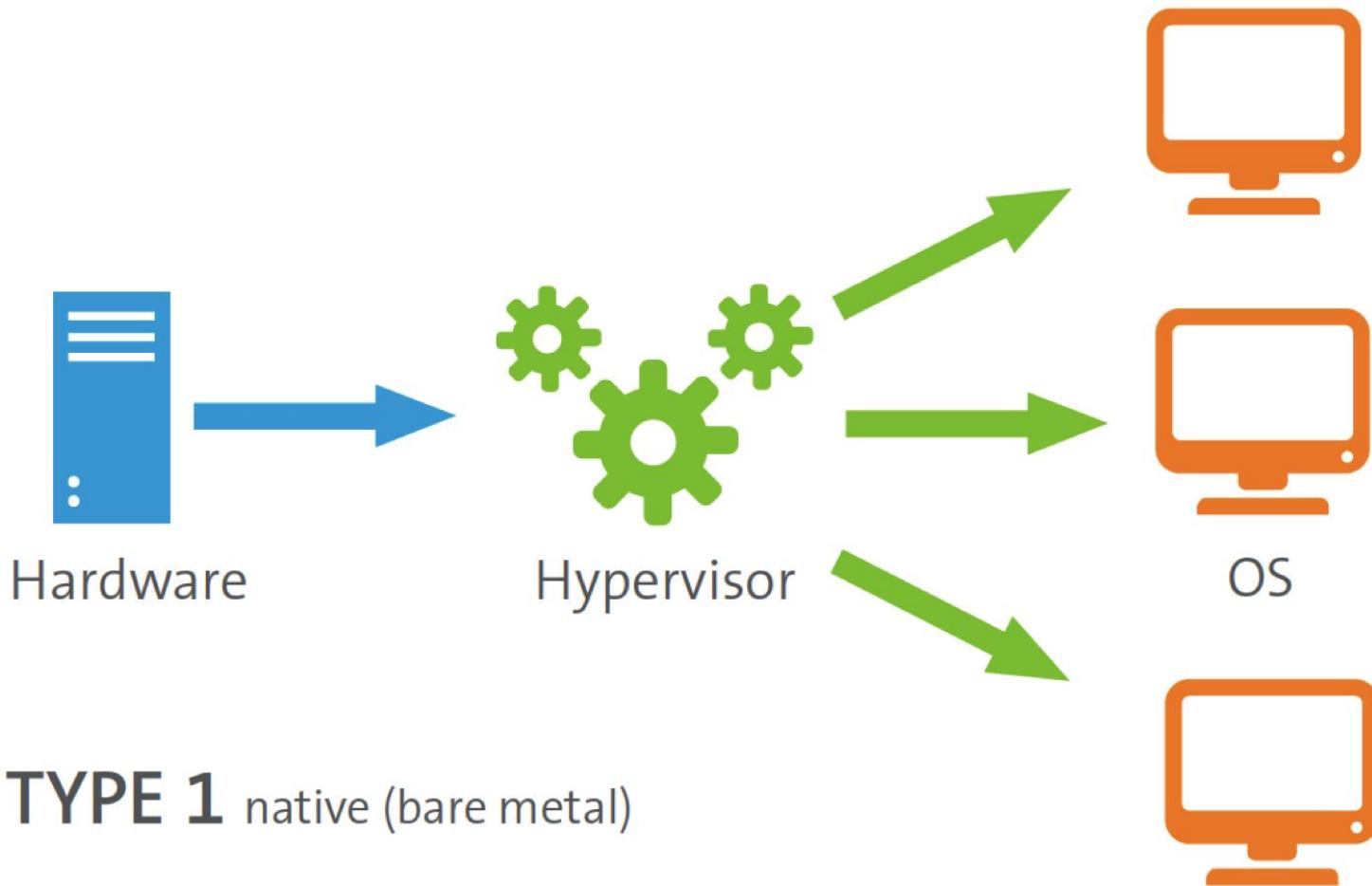
- **Tipos de VMs:**
- **Tipo 1 (Bare Metal):**
  - Executado diretamente no hardware físico.
  - Oferece maior desempenho e controle sobre os recursos.
  - Exemplo: **VMware ESXi, Microsoft Hyper-V.**
- **Tipo 2 (Hosted):**
  - Executado sobre um sistema operacional já existente.
  - Mais fácil de configurar e usar, mas com desempenho potencialmente inferior ao Tipo 1.
  - Exemplo: **Oracle VirtualBox, VMware Workstation.**

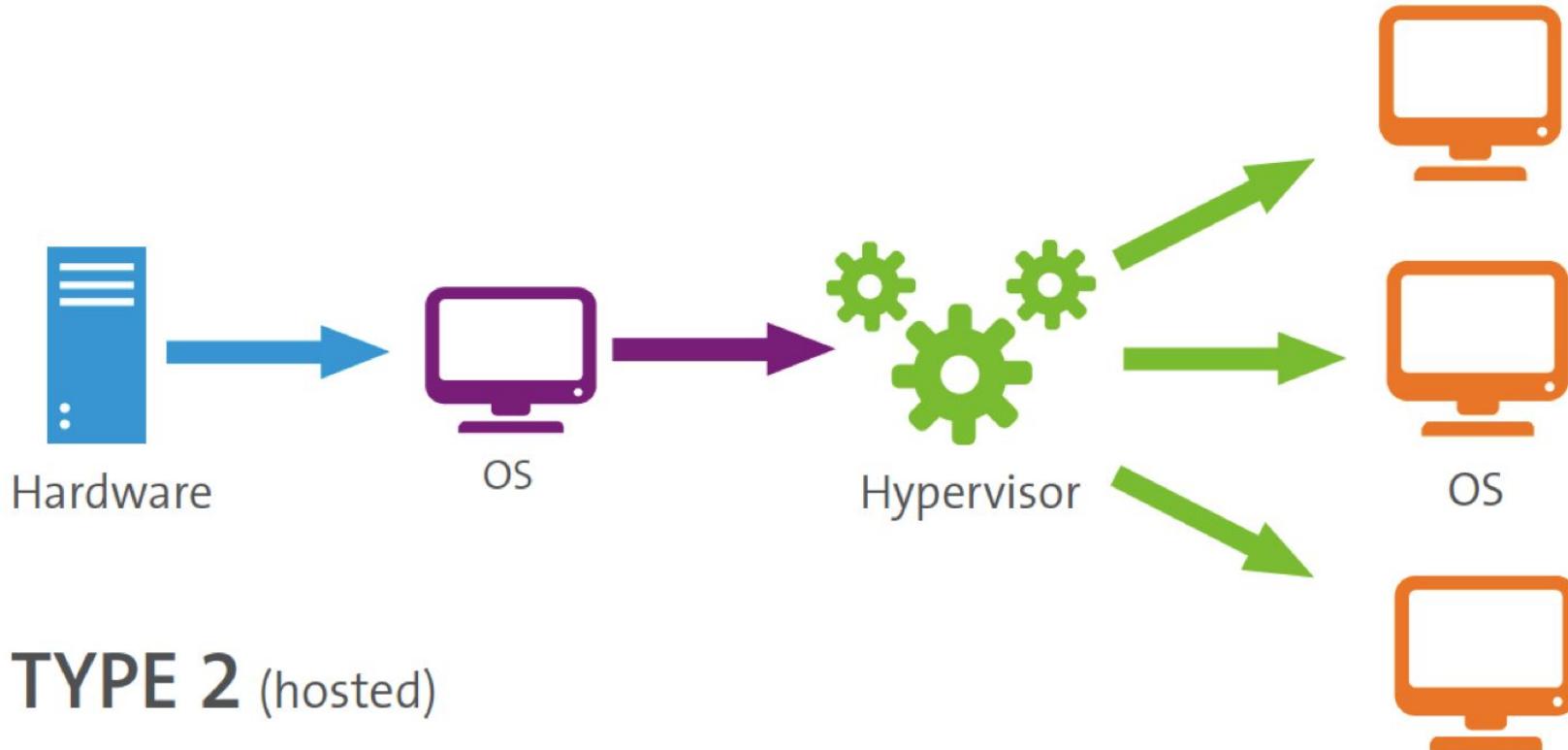


# Máquinas Virtuais (VMs)

- **Arquitetura: Hypervisor e Suas Funções:**
- **Hypervisor:**
  - Também conhecido como "**Monitor de Máquina Virtual**", é o software que permite a criação e execução de VMs.
  - **Gerencia e aloca** recursos físicos do host para cada VM, garantindo o **isolamento e o uso eficiente** dos recursos.
- **Funções do Hypervisor:**
  - **Isolamento:** Mantém as VMs separadas para evitar interferências.
  - **Gerenciamento de Recursos:** Aloca CPU, memória e armazenamento para as VMs conforme necessário.
  - **Facilidade de Backup e Recuperação:** Oferece funcionalidades para backup e migração de VMs com facilidade.
  - **Segurança:** Monitora e protege as VMs de ameaças e falhas.







# Máquinas Virtuais (VMs)

## Tipo I (Bare Metal):

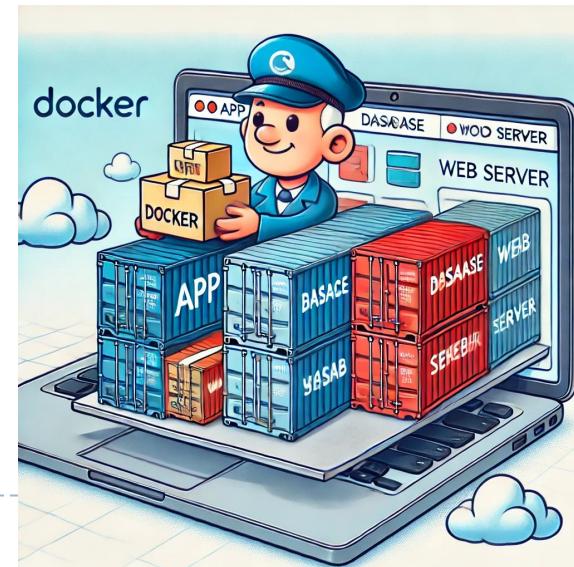
Desempenho Superior  
Eficiência de Recursos  
Segurança Maior  
Alta Escalabilidade  
Estabilidade Alta  
Configuração Complexa  
Custo Inicial Maior  
Menor Flexibilidade  
Compatibilidade Limitada

## Tipo 2 (Hosted):

Desempenho Inferior  
Eficiência de Recursos  
Moderada  
Segurança Menor  
Escalabilidade Menor  
Estabilidade Dependente  
Configuração Simples  
Custo Inicial Menor  
Alta Flexibilidade  
Compatibilidade Ampla

## • O que é Docker?

- Docker é uma plataforma de código aberto que automatiza a implantação de aplicações dentro de contêineres de software, proporcionando uma camada adicional de abstração e automação de virtualização em nível de sistema operacional.



# Doker

- **Jail (chroot)**

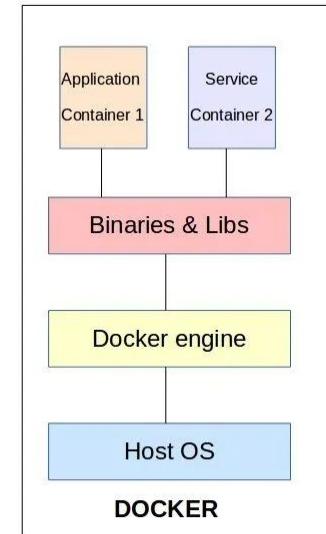
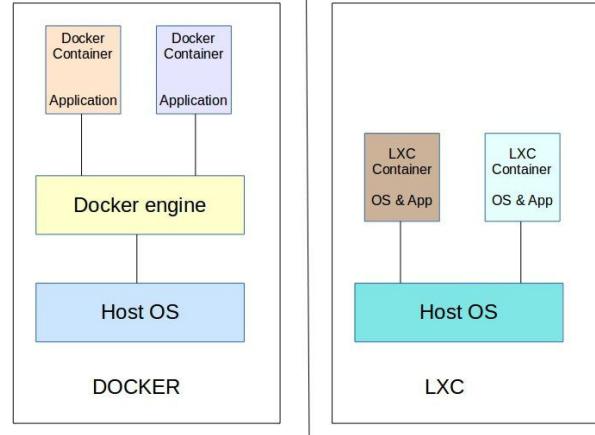
- Desde os primórdios \*unix possui a capacidade de isolar um processo e uma área do disco do restante do Sistema Operacional para que, em caso de falha ou invasão ao serviço, o acesso indevido ficasse restrito a área do disco e da memória definida no Jail.

- **LXC ( Linux Containers)**

- Possibilidade de isolar processos e área do disco no Linux, através de containers, desenvolvido em 2008 e posteriormente implantado no kernel do Linux (namespaces, cgroups, chroot, SELinux e apparmor).

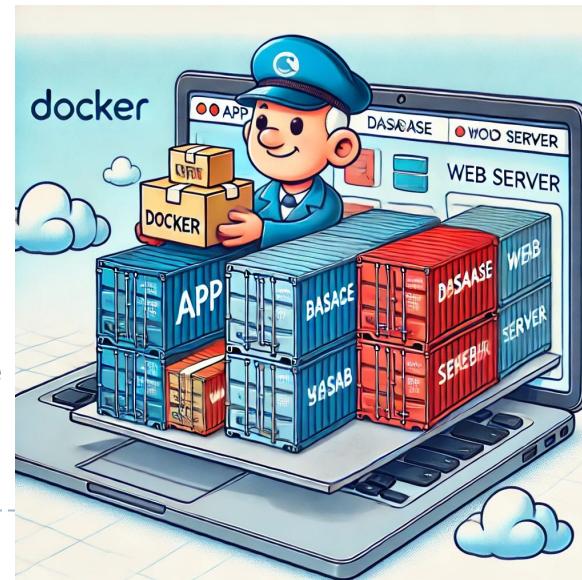
- **Docker**

- Criado em 2013, é uma evolução (ou não) do LXC, transformando o container em uma imagem, contendo tudo que é necessário para execução do serviço e possibilitando a execução do container em qualquer sistema hospedeiro (Host OS)



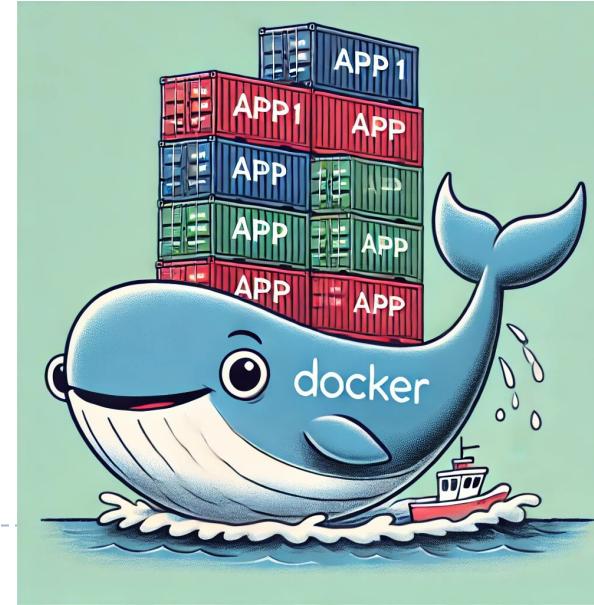
# Doker

- **Arquitetura:**
- **Imagens:**
  - Imagens Docker são os blocos de construção das contêineres. Elas são templates somente de leitura a partir dos quais os contêineres são criados.
- **Contêineres:**
  - Contêineres são instâncias em execução de imagens Docker que compartilham o kernel do sistema operacional, mas funcionam em processos isolados.
- **Docker Engine:**
  - Docker Engine é a aplicação cliente-servidor que cria e gerencia contêineres Docker. Inclui um daemon que é um processo de servidor de longa execução e uma interface de linha de comando (CLI).



# Doker vs VMs

- **Comparação com VMs:**
- **Isolamento em Nível de SO:**
  - Docker fornece isolamento ao nível do sistema operacional, o que significa que contêineres compartilham o mesmo kernel do SO, mas funcionam de maneira independente.
- **Isolamento em Nível de Hardware:**
  - VMs oferecem isolamento em nível de hardware completo, o que significa que cada VM inclui um sistema operacional completo e uma camada de virtualização para emular o hardware.



# Doker vs VMs

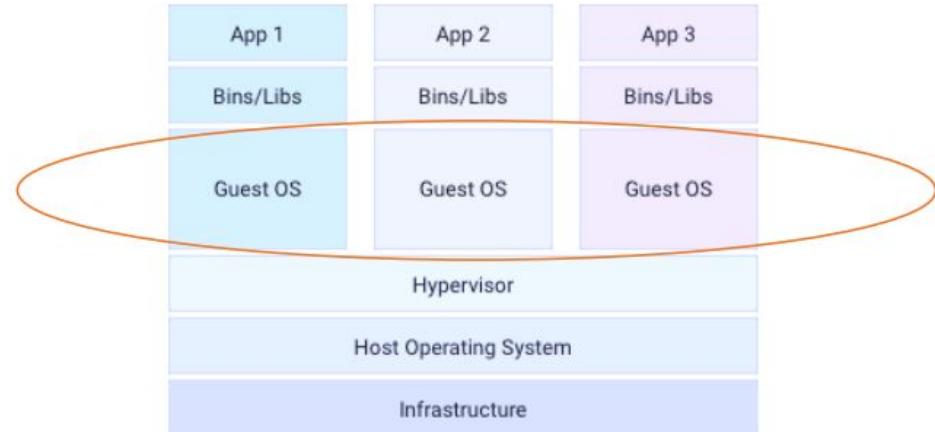
- **Containers**

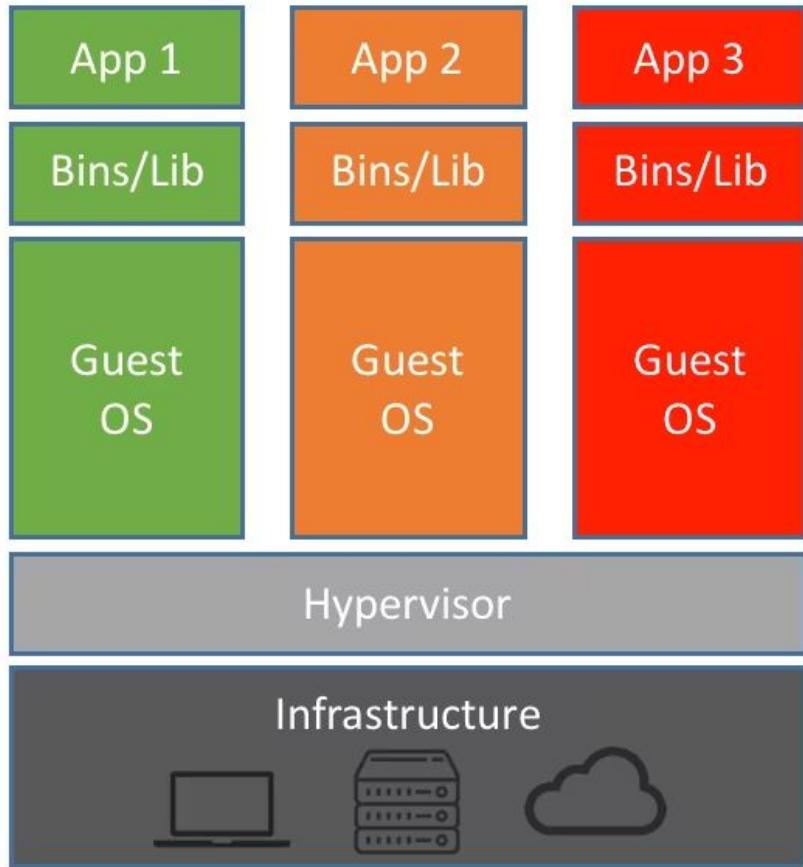
- Kernel do host compartilhado
- Portabilidade
- Escalabilidade mais rápida



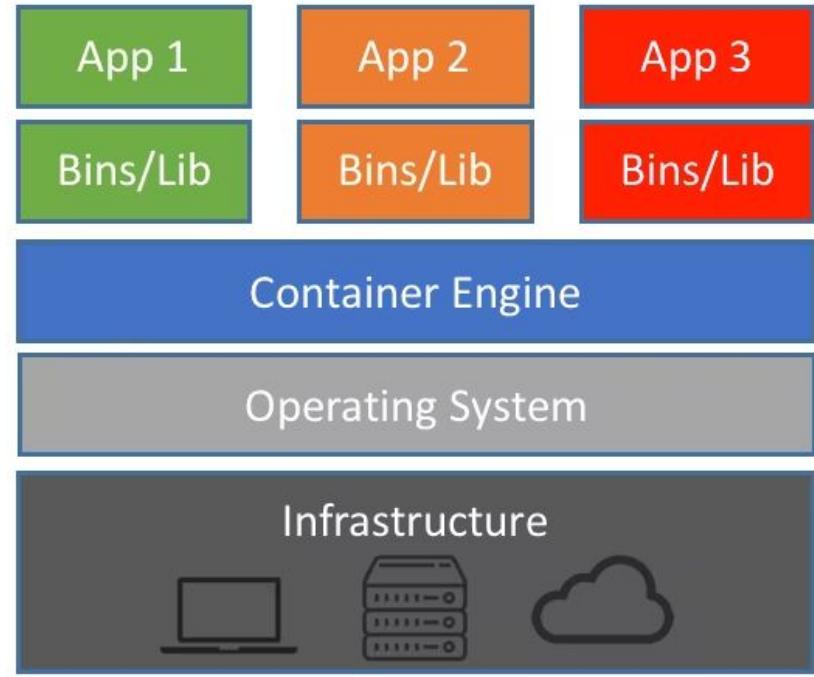
- **Virtual Machines**

- Sistema Operacional separado por instâncias





Machine Virtualization



Containers

# Doker

- Docker melhora a **eficiência do desenvolvimento e da implantação de software.**
- Contêineres são mais **leves e rápidos para iniciar e parar** em comparação com VMs.
- Docker facilita a **portabilidade de aplicações** entre diferentes ambientes de TI.
- **CI/CD:** Facilita a **integração contínua e a entrega contínua** (Continuous Integration/Continuous Delivery).
- **Microservices:** Ideal para arquiteturas de microserviços, onde cada serviço pode ser contido e gerenciado separadamente.
- Testes Automatizados: Permite a criação rápida de ambientes de teste automatizados.



## Vantagens:

**Leveza:** Contêineres são menos pesados e utilizam menos recursos que máquinas virtuais.

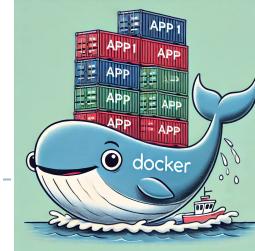
**Rapidez no Deployment:**  
Implementação rápida de aplicativos e serviços.

**Portabilidade:** Funciona de maneira consistente em diferentes ambientes, desde a máquina de desenvolvimento até a produção.

## Desvantagens:

**Segurança Compartilhada:**  
Contêineres compartilham o kernel do sistema operacional, o que pode aumentar os riscos de segurança.

**Isolamento Menor que VMs:** O isolamento não é tão robusto quanto o proporcionado por máquinas virtuais completas.



# Problema dos containers

Devido à quantidade de **aplicações/serviços** e a **granularidade** dos mesmos em **microserviços**, é necessária uma ferramenta que **gerencie** os containers.

- **Quais tem acesso externo?**
- **Quais portas estão abertas?**
- **Quantas instâncias de cada container está em execução?**
- **Como realizo um rollback de uma aplicação que está com problema?**
- **Como atualizo  $n$  instâncias de uma aplicação sem que ela saia de execução?**

**\*\* Resolver esses problemas é que a orquestração de containers promete \*\***

# Orquestração de containers - exemplos

---

- **Kubernetes**
  - Open Souce, desenvolvido pela Google e doado para o Cloud Native Computing Foundation
- **Docker Swarm**
  - Orquestrador de containers da Docker inc.
- **OpenShift**
  - Mantido pela Red Hat e também possui sua versão open source
- **AWS ECS - Elastic Container Service**
  - Baseado nos kubernetes
  - Integra com os demais serviços da Amazon DNS, balanceamento, segurança, monitoramento

# Kubernetes

- **Kubernetes**

- Provê a **orquestração dos containers**, de rede, dos volumes (discos) e dos demais recursos para que as aplicações executem com disponibilidade e escalabilidade
- **Não possui nenhum run-time nativo**: Precisa que seja instalado no mínimo um gerenciador de rede e o run-time dos containers (**Docker?**, **ContainerD?**, **CRI-O?**)
- O runtime é a camada que faz a **interface** entre o **contêiner** e o **sistema operacional do host**, gerenciando **recursos** como CPU, memória, rede e armazenamento.



# Kubernetes

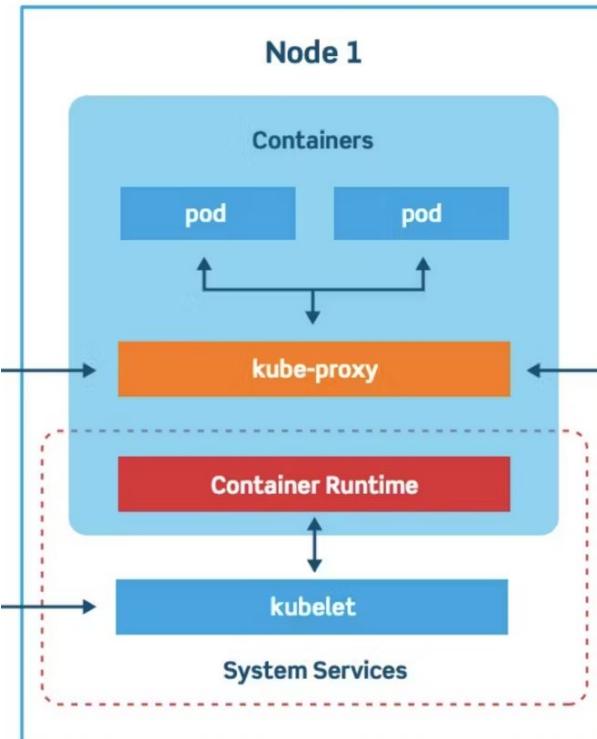
- Kubernetes é uma plataforma de código aberto para orquestração de contêineres, **criada pelo Google**, que automatiza a implantação, o dimensionamento e a **gestão de aplicativos em contêineres**.



# Kubernetes

- **Pods** são as unidades de trabalho que executam os contêineres.
- **Nodes** são as máquinas que hospedam e executam esses pods.
- **Runtime** é o software dentro de cada node que gerencia a criação, execução e remoção dos contêineres dentro dos pods.
- **Kubelet** é um agente que roda em cada node do cluster e é responsável por garantir que os contêineres dentro dos pods estejam sendo executados conforme desejado.
- **Kube-proxy** é responsável por gerenciar a rede do cluster e garantir que as comunicações de rede dentro do cluster funcionem corretamente.
- **Control Plane** a parte responsável por tomar decisões globais sobre o cluster, como agendar (schedule) os pods, além de detectar e responder a eventos do cluster.
- **Clusters** são conjuntos de nodes que trabalham juntos para fornecer uma plataforma de orquestração de contêineres.

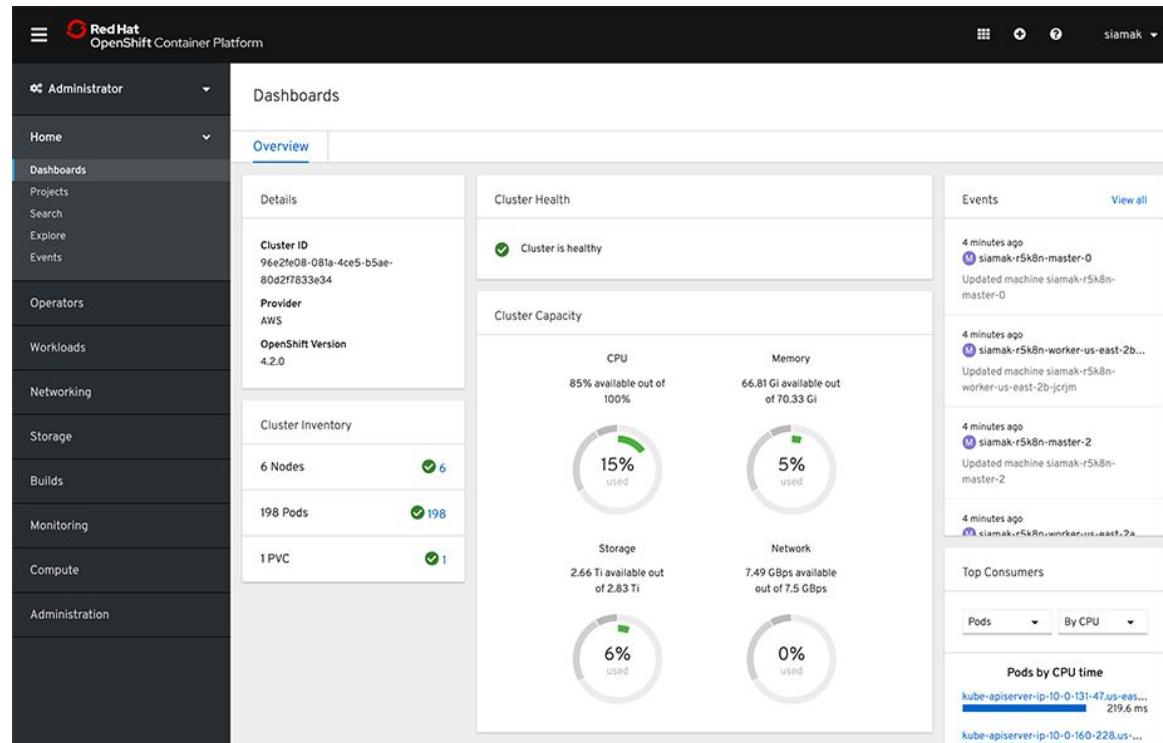
Worker Node Key Components



# Kubernetes

## OpenShift Console

- A interface gráfica baseada na web
- Visão Geral do Cluster
- Gerenciamento de Projetos
- Desenvolvimento e Deploy
- Monitoramento e Logs
- Integração Contínua (CI) e Implantação Contínua (CD)
- Gerenciamento de Usuários e Segurança
- Integração com Ferramentas DevOps



# Kubernetes

## Vantagens:

**Escalabilidade:** Kubernetes permite o escalonamento automático de aplicações, ajustando a quantidade de recursos com base na demanda, o que é essencial para lidar com cargas de trabalho variáveis.

**Automação:** Automatiza a implantação, gerenciamento e escalonamento de contêineres, reduzindo a necessidade de intervenção manual e permitindo operações mais ágeis.

**Resiliência:** Proporciona alta disponibilidade e recuperação automática de falhas, garantindo que os serviços estejam sempre operacionais mesmo em caso de problemas.

## Desvantagens:

**Complexidade:** Kubernetes pode ser complexo de configurar e gerenciar, especialmente para iniciantes ou para empresas sem uma equipe de TI dedicada.

**Curva de Aprendizado:** A curva de aprendizado pode ser íngreme devido à ampla gama de funcionalidades e à necessidade de entender conceitos como pods, nodes, e clusters.

# Computação em Nuvem

- A **computação** vem evoluindo de maneira absurda nas últimas décadas. E essa evolução se dá de tal forma, que o ser humano agora implementa tecnologia até nas nuvens!

[LINK](#)



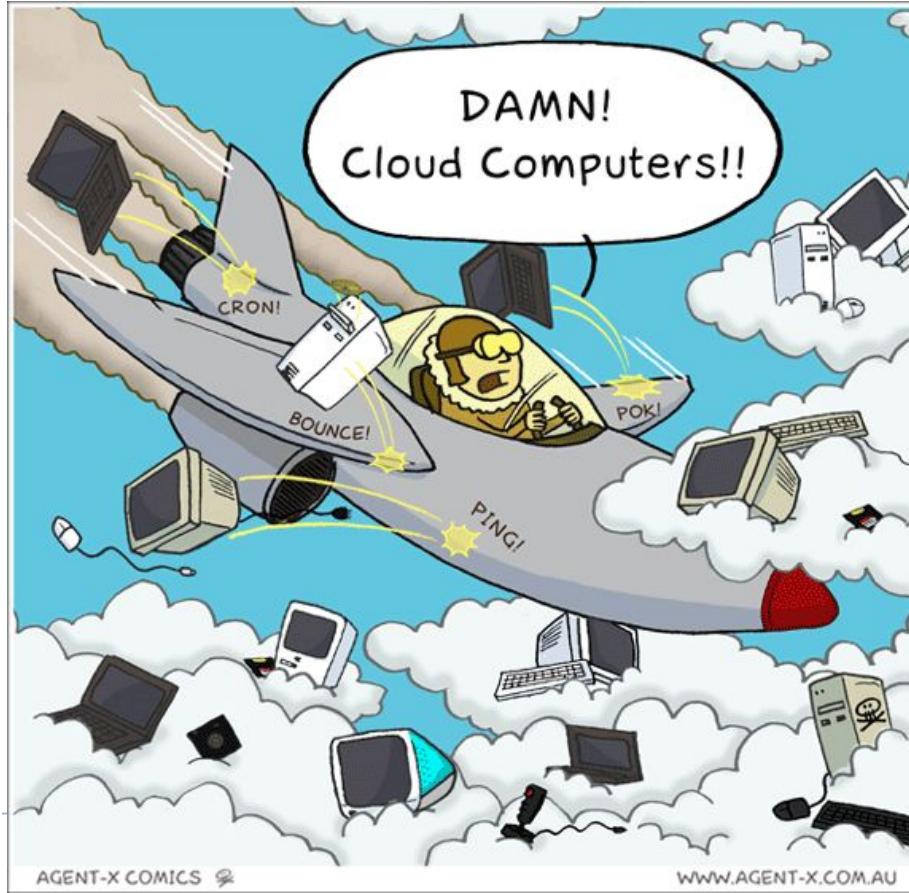
# O que é computação em Nuvem?

Atualmente, muito se fala sobre computação em nuvem:

- "Meus dados estão subindo para o Dropbox!"  
ou
- "Minha empresa está usando a nuvem como ferramenta de escritório para uso no dia a dia!"  
ou
- "Somos uma pequena empresa e utilizamos uma infraestrutura na nuvem"



# O que é computação em Nuvem?

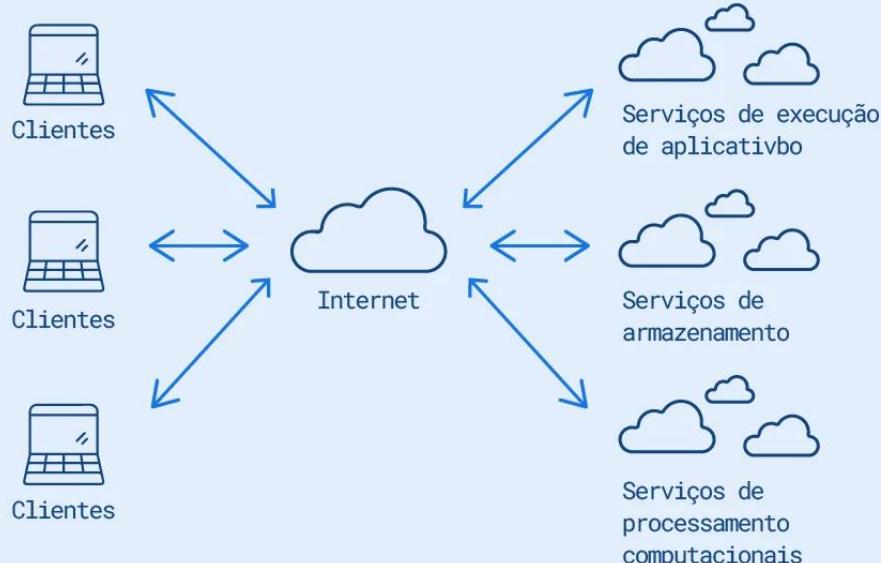


# Afinal, o que é essa “nuvem” ou Cloud?

- Uma nuvem pode ser entendida como um **conjunto de aplicações, armazenamento e computação** que tem a **internet** como base e plataforma de funcionamento.
- E esse pool de serviços possui capacidade suficiente para dar suporte à maioria das necessidades de grande parte dos usuários.

# Afinal, o que é essa “nuvem” ou Cloud?

## \_Definindo o que é Cloud



Segundo uma definição do NIST (National Institute of Standards and Technology), referenciada no livro “Web Services RESTful” (2015), de Ricardo Lecheta:

“Cloud Computing é um modelo de computação que provê um conjunto compartilhado de recursos de computação customizáveis como redes, servidores, armazenamento, aplicações e serviços.”

# Qual é a função da nuvem?

- Imagine que você está desenvolvendo um relatório importante com o seu colega de trabalho, em seu próprio computador e o PC desliga.



# Qual é a função da nuvem?



# Qual é a função da nuvem?

- No entanto, imagine agora que você está utilizando um armazenamento em nuvem com um backup automático. Isso quer dizer que o seu arquivo estava sendo salvo continuamente na nuvem, dessa forma, você pode utilizar outra máquina para recuperar o arquivo atualizado e retomar o trabalho de onde você parou.
- **Armazenamento de dados**

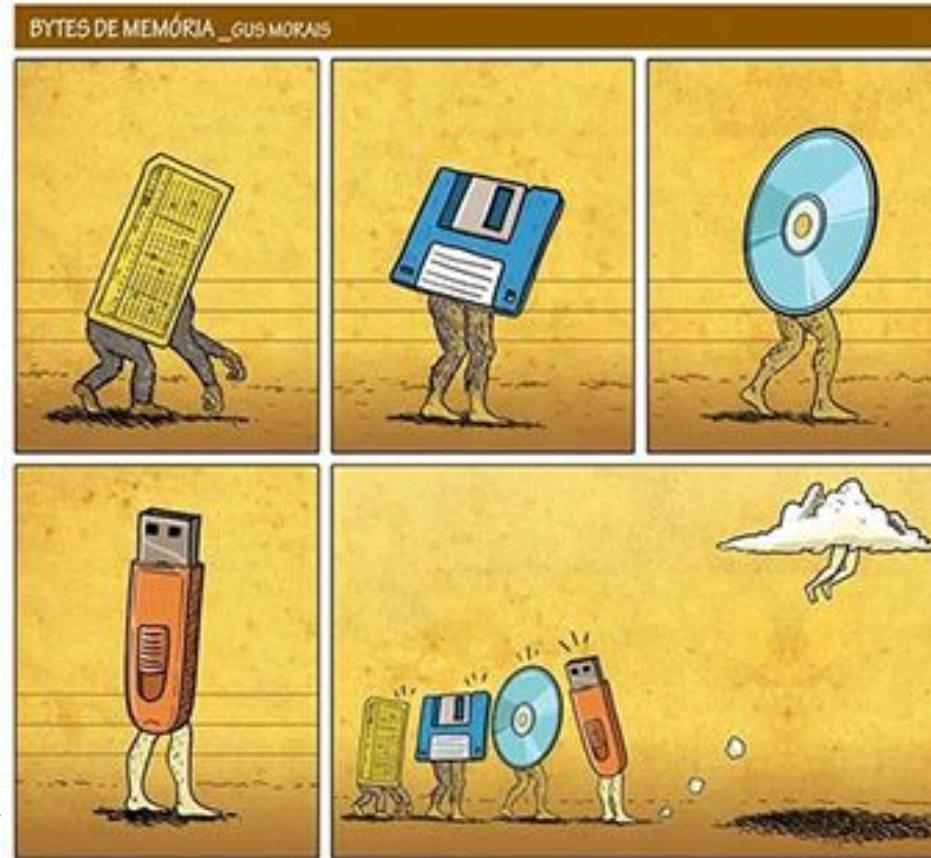


# Qual é a função da nuvem?

- Armazenamento de dados
- compartilhamento de arquivos
- Backups e recuperação de dados
- Google, Microsoft, Amazon e Dropbox



# Qual é a função da nuvem?



# Qual é a função da nuvem?

- Google Docs, Google Drive, Microsoft OneDrive, iCloud, Trello:
- Essas ferramentas permitem que as pessoas **criem, accessem, editem e colaborem** em arquivos como **documentos, fotos, planilhas e apresentações**, enquanto sincroniza e salva os documentos em tempo real na nuvem.
- **Conexão à Internet**



# Qual é a função da nuvem?

- **Hospedagem de aplicativos**
  - Gerenciar os aplicativos
  - Não é necessária a configuração e manutenção de servidores físicos
- **Processamento de dados**
  - Executar tarefas computacionais de processamento
  - Análise e armazenamento de grandes volumes de dados.



# Onde ficam os arquivos na nuvem?



# Onde ficam os arquivos na nuvem?

- Em **servidores remotos** que são mantidos por **provedores** de serviços em nuvem.
- Esses provedores **investem em infraestrutura robusta de data centers** em **diferentes localizações geográficas**, que garantem a **disponibilidade** de recursos de computação, **armazenamento, segurança e confiabilidade** dos dados a serem adicionados na nuvem.



# Onde ficam os arquivos na nuvem?

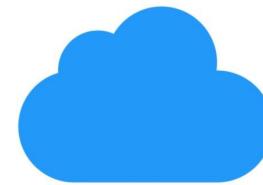
## NÃO EXISTE NUVEM



É APENAS UM MONTE DE  
SERVIDORES LINUX

# Onde ficam os arquivos na nuvem?

- Sistema Operacional Android - Padrão Google
- Assim como a Apple é o provedor de serviços em nuvem padrão para o iOS, iCloud
- Mesmo que o Google Cloud e o iCloud sejam disponibilizados para ambos os sistemas operacionais.



iCloud



Google Drive

# O que é um aplicativo Cloud?

- Essas aplicações são softwares executados em servidores remotos na nuvem, diferente dos aplicativos tradicionais que necessitam ser instalados e executados localmente em nossos computadores pessoais ou smartphones.
- **Dentre as diversas razões para utilizar esses aplicativos, podemos citar:**
  - A acessibilidade e flexibilidade
  - Escalabilidade
  - Redução de custos
  - Ambientes colaborativos

# **Exemplos do uso de Cloud no dia a dia**

---

- **Gmail, Outlook**
- **Google Drive, Dropbox, iCloud**
- **Netflix, Spotify, YouTube**
- **Microsoft Teams, Slack, Zoom**
- **Google Stadia, Xbox Cloud Gaming**
- **Educação e E-learning**
- **WordPress, Wix**
- **GitHub, GitLab**
- **Salesforce, SAPAmazon Alexa, Google Assistant**

- É interessante ressaltar também que, com o surgimento do termo “computação em nuvem”, **uma visão da oferta de serviços de infraestrutura física ou virtual** passou a ser realizada por meio de **software**, normalmente **pago**, o que reduz e **terceiriza a infraestrutura** do usuário.

# Quem pode usar o Cloud?



# Modelos de Cloud

## \_Modelos de Cloud

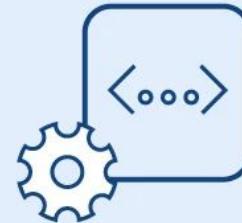
IaaS

Infrastructure  
as a Service



PaaS

Platform  
as a Service



SaaS

Software  
as a Service



# IaaS - Infrastructure as a Service

- A **Infraestrutura como Serviço** oferece recursos de computação, armazenamento e rede. Ou seja, esse modelo disponibiliza um ambiente com uma infraestrutura virtualizada.
- **Como exemplos de serviços IaaS:**
  - Amazon Web Services;
  - Google Compute Engine.



# PaaS - Plataform as a Service

- Na **Plataforma como Serviço**, temos um programa ou aplicativo entregue com o objetivo de facilitar a implementação de algum serviço; ou seja, é criado um ambiente para a programação e o desenvolvimento de alguma solução.

- **Alguns exemplos de PaaS são:**

- Heroku;
- Red Hat OpenShift.
- **IBM Cloud Foundry** fornece aos desenvolvedores uma maneira de construir, testar, implantar e escalar



# SaaS - Software as a Service

- No **Software como Serviço**, o software é executado em um servidor, de forma que o cliente não precisa tê-lo instalado em sua máquina.
- **Como exemplos de SaaS, encontramos ferramentas como:**
  - **Gupy**, plataforma online de recrutamento e seleção;
  - **Trello**, plataforma utilizada para gerenciamento de tarefas e projetos.



# Tipos de Cloud



# Cloud Pública

- Este sem dúvida é o tipo de serviço de cloud que mais vemos em utilização pelas empresas.
  - Nele, os serviços são **mantidos por um determinado provedor de serviços**.
  - Esses provedores possuem uma **série de datacenters** com uma gama de servidores interconectados que compartilham recursos e serviços pelos utilizadores da nuvem.
- 
- **Exemplos:**
    - Amazon Web Services (AWS)
    - Microsoft Azure:
    - Google Cloud Platform (GCP):



# Cloud Privada

- Este tipo de serviço oferece uma infraestrutura alocada para uma única organização. Uma nuvem privada pode estar hospedada no data center mantido pela própria empresa.

- **Exemplos:**

- VMware vCloud
- Microsoft Azure Stack
- OpenStack



# Cloud Híbrida

- Uma nuvem híbrida é a utilização em conjunto da nuvem pública e da privada.
- Um cenário muito comum seria o de usar **uma nuvem pública para manter a execução de algum sistema e manter a fonte de dados da aplicação na nuvem privada.**



# On Premise

- Quando estamos falando do ambiente de nuvem, podemos nos deparar com o termo **On Premise**, que é um tipo de organização da nossa **estrutura em um ambiente só**.
- Um ambiente On Premise é quando temos **toda a infraestrutura de servidores e máquinas em um espaço físico na empresa**, a famosa “**sala de servidores**” ou datacenter.



# Vantagens do On-Premise

- **Maior controle e personalização** – Como a empresa é dona da infraestrutura, pode configurá-la como quiser.
- **Segurança e conformidade** – Ideal para empresas que precisam atender a requisitos rigorosos de governança de dados.
- **Independência da Internet** – Como os sistemas estão dentro da própria organização, não dependem de conexão externa para funcionamento.

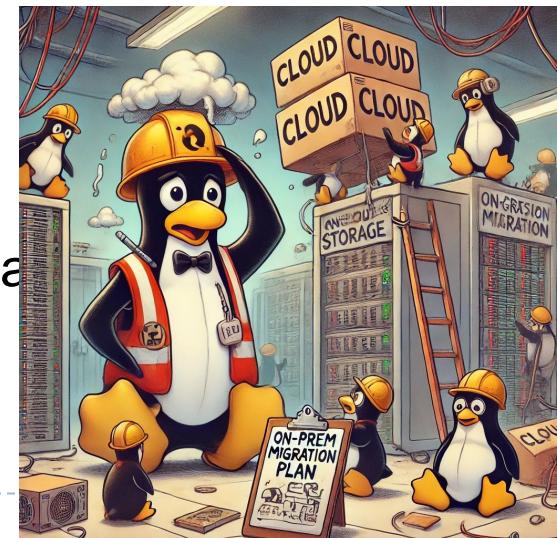


# Desvantagens do On-Premise

⚠ **Custo elevado** – Desde a aquisição de hardware até a manutenção contínua, os gastos podem ser muito altos.

⚠ **Menos flexibilidade** – A escalabilidade exige compra de novos equipamentos, tornando o processo lento e caro.

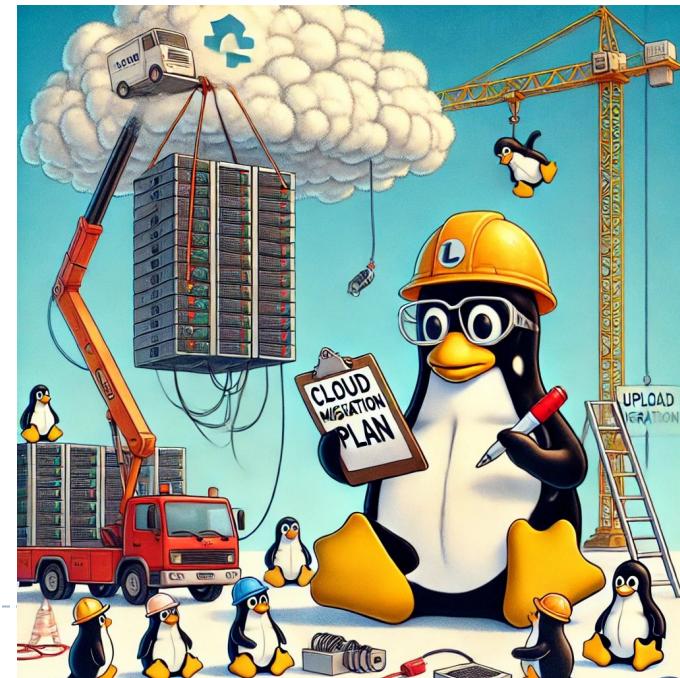
⚠ **Gestão complexa** – Requer equipe especializada para monitorar, atualizar e proteger a infraestrutura.



# On-Premise vs. Computação em Nuvem



- ◆ **On-Premise:** Controle total, alto custo inicial, manutenção própria, escalabilidade limitada.
- ◆ **Cloud Computing:** Custo baseado no uso, escalabilidade sob demanda, manutenção feita pelo provedor.



# Conclusão



- O modelo **On-Premise** ainda é utilizado por empresas que precisam de segurança rígida, baixa latência e total controle da infraestrutura.
- No entanto, cada vez mais organizações estão migrando para a **nuvem** devido à flexibilidade, escalabilidade e redução de custos operacionais.





**Uni**senac  
Centro Universitário RS

 **Senac** Fecomércio Sesc

MUITO OBRIGADO!!!!

Guto Muniz

[augustomuniz@gmail.com](mailto:augustomuniz@gmail.com)